Mood Manifolds: A Neuromodulatory Control Surface for Deep Reinforcement Learning

Dario Fumarola Jin Tan Ruan
Amazon Web Services

Abstract

Deep reinforcement learning agents are usually trained around one reward and safety trade-off. Deployment rarely stays at that operating point: the same policy may need to collect reward aggressively, avoid nearby hazards, or explore when evidence is scarce. We study a three-dimensional control surface over reward gain, hazard-cost gain, and action temperature. The resulting agent, MC-PPO, trains one actor-critic policy conditioned on a mood vector $m = (\beta_R, \beta_C, \tau)$; after training, changing m changes the action distribution while every network weight remains fixed. The architecture uses FiLM modulation in the policy trunk, separate reward and cost critics for stable scalarization, and a monotone temperature gate on the actor logits. In Pac-Mind++ and MiniHack-HazardRooms, one frozen policy traces a reward-risk frontier, interpolates to held-out moods, and reroutes behavior after a hazard alarm. Negative controls show why this has to be a deployment input rather than only a loss weight: a policy that never reads m cannot be steered by rewriting m after training. The contribution is a compact interface for runtime retargeting, not a literal model of neuromodulator biology.

1 Introduction

A reinforcement learning policy is often shipped with the operating point it learned: one reward function, one safety penalty, one entropy schedule. When the operating point moves, the usual remedies are heavy. One can fine-tune, train a specialist, or select from a library. None of these is a natural interface for situations in which context changes inside an episode. A household robot may move quickly in an empty room and slowly near a child. A game agent may harvest resources until an enemy appears, then give up reward for survival. A navigation policy may accept risk when time is scarce and avoid it when the environment is crowded. The dynamics are the same; the valuation of trajectories changes.

Biological control systems offer one useful design pattern. Neuromodulators broadcast low-bandwidth signals that retune large circuits without replacing them. Computational neuroscience has connected these signals to reinforcement-learning quantities such as reward prediction error, punishment, exploration, and learning rate [Schultz et al., 1997, Doya, 2002, Daw et al., 2002]. We do not try to reproduce that biology. We take the engineering lesson: a trained agent should expose a small set of meaningful inputs through which another controller can change reward seeking, caution, and exploration at runtime.

We define the control input as

$$m = (\beta_R, \beta_C, \tau), \tag{1}$$

where β_R scales appetitive reward, β_C scales hazard cost, and τ controls the action temperature. The deployed actor is $\pi_\theta(a \mid o, m)$ rather than $\pi_\theta(a \mid o)$. This distinction drives the whole paper. A scalar that only multiplies losses during training may affect the checkpoint that optimization finds, but it has no runtime semantics unless the frozen computation that produces actions also reads that scalar.

We implement this idea as Mood-Conditioned PPO (MC-PPO). The policy encodes the observation once, passes the representation through FiLM-conditioned layers, and produces logits whose final temperature is a monotone function of τ [Perez et al., 2018]. The critic predicts reward and cost returns with separate heads, then combines them according to the current mood. The separation avoids folding reward and safety into one value target too early and makes the method close to preference-conditioned multi-objective control [Schaul et al., 2015, Barreto et al., 2017, Hayes et al., 2022, Reymond et al., 2022].

The experiments test the interface directly. We freeze the network, sweep moods that were not used as anchors, and measure return, cost, entropy, local policy sensitivity, and interpolation error. We also trigger a mid-episode switch from a reward-seeking mood to a cautious mood when the cost critic raises an alarm. The resulting behavior is compared with neutral PPO, fixed-limit Lagrangian PPO, concatenation conditioning, FiLM ablations, loss-only modulation, and a 64-policy specialist library.

The main result is that the low-dimensional input behaves like a usable control surface rather than an arbitrary side channel. MC-PPO covers most of the specialist frontier with one policy, has smoother transitions between operating points, and retains enough separation between moods to support online switching. The loss-only baseline is important because it rules out a common shortcut: optimizer-side gains can produce different training runs, but they cannot control a frozen actor that omits the gain from its forward pass.

2 Related Work

Neuromodulation and reinforcement learning. Dopamine has a canonical link to reward prediction error through both computational models and neural recordings [Schultz et al., 1997]. Doya's metalearning account assigned different global reinforcement-learning parameters to ascending neuromodulatory systems, including temporal-difference error, dis-

counting, action randomness, and learning rate [Doya, 2002]. Daw, Kakade, and Dayan analyzed opponent serotonin and dopamine signals in reward and punishment statistics [Daw et al., 2002]. Our variables use that vocabulary only as a naming guide. β_R is a reward gain, β_C is a cost gain, and τ is an action-temperature coordinate.

Entropy, safety, and costs. Maximum-entropy RL puts entropy inside the control objective, with Soft Actor-Critic as a standard off-policy example [Haarnoja et al., 2018]. Safe RL usually separates task reward from safety cost through constrained MDPs, Lagrangian updates, or trust-region constraints [Achiam et al., 2017]. MC-PPO borrows the reward-cost decomposition but changes the target of optimization: instead of solving for one safe optimum, it learns a family of operating points indexed by a small control vector.

Conditioned and multi-objective policies. Universal value-function approximators condition value estimates on goals [Schaul et al., 2015]. Successor features factor dynamics from rewards so that policies can transfer across reward changes [Barreto et al., 2017]. Multi-objective RL studies vector-valued returns and preference-conditioned policies, including Pareto Conditioned Networks [Hayes et al., 2022, Reymond et al., 2022]. MC-PPO sits in this family. Its preference vector is chosen for deployment control rather than exhaustive Pareto coverage.

Evaluation discipline in deep RL. Actor-critic results depend on seeds, implementation details, and aggregation choices [Henderson et al., 2018]. We report intervals over seeds and held-out layouts and use stratified bootstrap confidence intervals for aggregate statistics, following recent recommendations for RL evaluation [Agarwal et al., 2021]. The aim is to evaluate the shape of a control surface, not a single best checkpoint.

3 Problem Formulation

We consider a discounted Markov decision process with costs,

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, c, \gamma), \tag{2}$$

where $r(s, a, s') \in \mathbb{R}$ is task reward, $c(s, a, s') \in \mathbb{R}_{\geq 0}$ is hazard cost, and $\gamma \in [0, 1)$ is the discount. The agent observes $o_t = \Omega(s_t)$ and samples

$$a_t \sim \pi_\theta(\cdot \mid o_t, m), \qquad m = (\beta_R, \beta_C, \tau) \in \mathcal{M}_{\text{mood}}. \tag{3}$$

For a fixed mood, the scalarized objective is

$$J(\theta; m) = \mathbb{E}_{\pi_\theta}\left[\sum_{t\geq 0} \gamma^t (\beta_R r_t - \beta_C c_t)\right] + \tau\, \mathbb{E}_{\pi_\theta}\left[\sum_{t\geq 0} \gamma^t H(\pi_\theta(\cdot \mid o_t, m))\right]. \tag{4}$$

The axes specify how trajectories are ranked. They do not assume the learned effects are independent. In practice, a high temperature can change state visitation and therefore cost; a high cost gain can reduce reward by steering around hazards.

Definition 1 (Deployment control)**.** *A frozen policy $\pi_{\theta^\star}$ has deployment mood control on a set $\mathcal{M}_{\text{mood}}$ if there exist an observation o and moods m, m' such that*

$$\pi_{\theta^\star}(\cdot \mid o, m) \neq \pi_{\theta^\star}(\cdot \mid o, m'). \tag{5}$$

The condition is evaluated with fixed parameters. It is not satisfied by training a new checkpoint for each mood.

We use four measurements. The reward-risk frontier is the set of pairs $(J_R, -J_C)$ obtained by sweeping m after freezing the network. Local sensitivity is

$$S_{\text{loc}} = \mathbb{E}_{o,m,\epsilon}\left[\frac{D_{\text{KL}}(\pi_\theta(\cdot \mid o, m) \| \pi_\theta(\cdot \mid o, m + \epsilon))}{\|\epsilon\|_2^2}\right] \tag{6}$$

computed on a shared observation buffer. Held-out interpolation error compares outcomes at unseen moods with multilinear interpolation from neighboring evaluated grid moods. Hypervolume (HV) summarizes the normalized frontier over expected return and negative expected cost.

4 Mood-Conditioned Actor-Critic

4.1 Architecture

MC-PPO uses a shared encoder $f_\psi(o)$ followed by a FiLM-conditioned trunk. For hidden layer ℓ,

$$h_{\ell+1} = \sigma\left(\gamma_\ell(m) \odot \text{LN}(W_\ell h_\ell + b_\ell) + \eta_\ell(m)\right), \tag{7}$$

where γ_ℓ and η_ℓ are produced by a small mood MLP $g_\phi(m)$. The FiLM gains are initialized near identity, so early training behaves like an ordinary actor-critic.

The actor head produces logits $z_\theta(o, m)$. The temperature coordinate is mapped monotonically to a positive logit temperature,

$$T(\tau) = T_{\min} \exp\{\tau(\log T_{\max} - \log T_{\min})\}, \qquad \tau \in [0, 1], \tag{8}$$

then

$$\pi_\theta(a \mid o, m) = \text{softmax}\left(z_\theta(o, m)_a / T(\tau)\right). \tag{9}$$

This gate gives τ a direct runtime effect. The FiLM path can still learn state-dependent exploration behavior, but increasing τ cannot reduce the final temperature.

The critic has separate heads $V_R(o, m)$ and $V_C(o, m)$ for reward and cost returns. PPO uses their scalarized combination

$$V_m(o) = \beta_R V_R(o, m) - \beta_C V_C(o, m). \tag{10}$$

Keeping the heads separate stabilizes learning when the same state is good under one preference and bad under another.

4.2 Training objective

Moods are sampled per episode and kept fixed until the environment terminates. If a rollout segment cuts through an unfinished episode, the same mood continues in the next segment. We compute generalized advantages separately:

$$\delta_t^R = r_t + \gamma V_R(o_{t+1}, m) - V_R(o_t, m), \tag{11}$$

$$\delta_t^C = c_t + \gamma V_C(o_{t+1}, m) - V_C(o_t, m), \tag{12}$$

$$\hat{A}_t^R = \sum_{l \geq 0} (\gamma\lambda)^l \delta_{t+l}^R, \qquad \hat{A}_t^C = \sum_{l \geq 0} (\gamma\lambda)^l \delta_{t+l}^C, \tag{13}$$

$$\hat{A}_t^m = \beta_R \hat{A}_t^R - \beta_C \hat{A}_t^C. \tag{14}$$

The clipped PPO term is

$$L_{\text{clip}} = \mathbb{E}_t \left[\min\{\rho_t \hat{A}_t^m, \text{clip}(\rho_t, 1-\epsilon, 1+\epsilon)\hat{A}_t^m\} \right], \tag{15}$$

where $\rho_t = \pi_\theta(a_t \mid o_t, m)/\pi_{\theta_{\text{old}}}(a_t \mid o_t, m)$. The maximized objective is

$$\begin{aligned} L = L_{\text{clip}} &+ \tau \mathbb{E}_t[H(\pi_\theta(\cdot \mid o_t, m))] \\ &- \alpha_R L_{V_R} - \alpha_C L_{V_C} - \alpha_M R_M. \end{aligned} \tag{16}$$

The mood regularizer is applied on a replayed observation batch:

$$\begin{aligned} R_M = \lambda_{\text{sm}} \mathbb{E} &\left[\frac{D_{\text{KL}}(\pi_m \| \pi_{m+\nu})}{\|\nu\|_2^2} \right] \\ &- \lambda_{\text{sep}} \mathbb{E}[\min(D_{\text{JS}}(\pi_m, \pi_{\bar{m}}), d_0)]. \end{aligned} \tag{17}$$

The first term discourages sharp local folds; the second prevents all moods from collapsing to the same action distribution.

Algorithm 1 Mood-Conditioned PPO

```
Initialize encoder, actor, critic, and mood-MLP parameters.
for each PPO iteration do
    for each parallel environment that resets do
        Sample an episode mood m ~ p_train(m).
    end for
    Roll out π_θ(a | o, m) for T steps, carrying each mood across rollout boundaries until reset.
    Estimate Â^R and Â^C with GAE.
    Form Â^m = β_R Â^R − β_C Â^C.
    Update actor and critics for K PPO epochs with the clipped objective and decomposed value losses.
    Apply the mood smoothness and separation terms on a shared observation batch.
end for
Freeze all weights. At deployment, update only m.
```

4.3 Elementary properties

The policy-gradient estimator is the ordinary scalarized estimator written in decomposed form.

Proposition 1 (Scalarized policy gradient)**.** *For fixed $m = (\beta_R, \beta_C, \tau)$ and policy $\pi_\theta(\cdot \mid \cdot, m)$, replacing the advantage in the policy-gradient estimator with $A_m = \beta_R A_R - \beta_C A_C$ gives the policy gradient of the scalarized reward-cost objective, with the entropy gradient added separately.*

Proposition 2 (Fixed-policy Lipschitz value)**.** *Assume $|r_t| \leq R_{\max}$, $0 \leq c_t \leq C_{\max}$, and finite $\mathcal{A}$. For any fixed policy π and two moods m, m',*

$$\begin{aligned} |J_\pi(m) - J_\pi(m')| \leq \frac{R_{\max}|\Delta_R| + C_{\max}|\Delta_C|}{1-\gamma} \\ + \frac{\log|\mathcal{A}|\,|\Delta_\tau|}{1-\gamma}. \end{aligned} \tag{18}$$

This statement concerns the value of a fixed policy; smoothness of a learned neural policy must be measured.

Proposition 3 (No runtime control without conditioning)**.** *Let a frozen deployed policy be $\pi_\theta(a \mid o)$, and let m appear only in the loss used before training ends. Then $\pi_\theta(\cdot \mid o, m) = \pi_\theta(\cdot \mid o, m')$ for all o, m, m'.*

4.4 Deployment calibration

The learned surface is treated as an interface with an admissible set, not as permission to choose arbitrary gains. Before online use, we evaluate a validation grid over $\mathcal{M}_{\text{mood}}$, record return-cost envelopes, estimate local KL between neighboring moods, and mark regions where cost rises faster than the monitor can react. The same grid defines a maximum allowed step size for transitions in m. These checks are empirical rather than formal guarantees, but they make the supervisor's choices visible: the controller moves on a measured chart instead of switching among unrelated checkpoints.

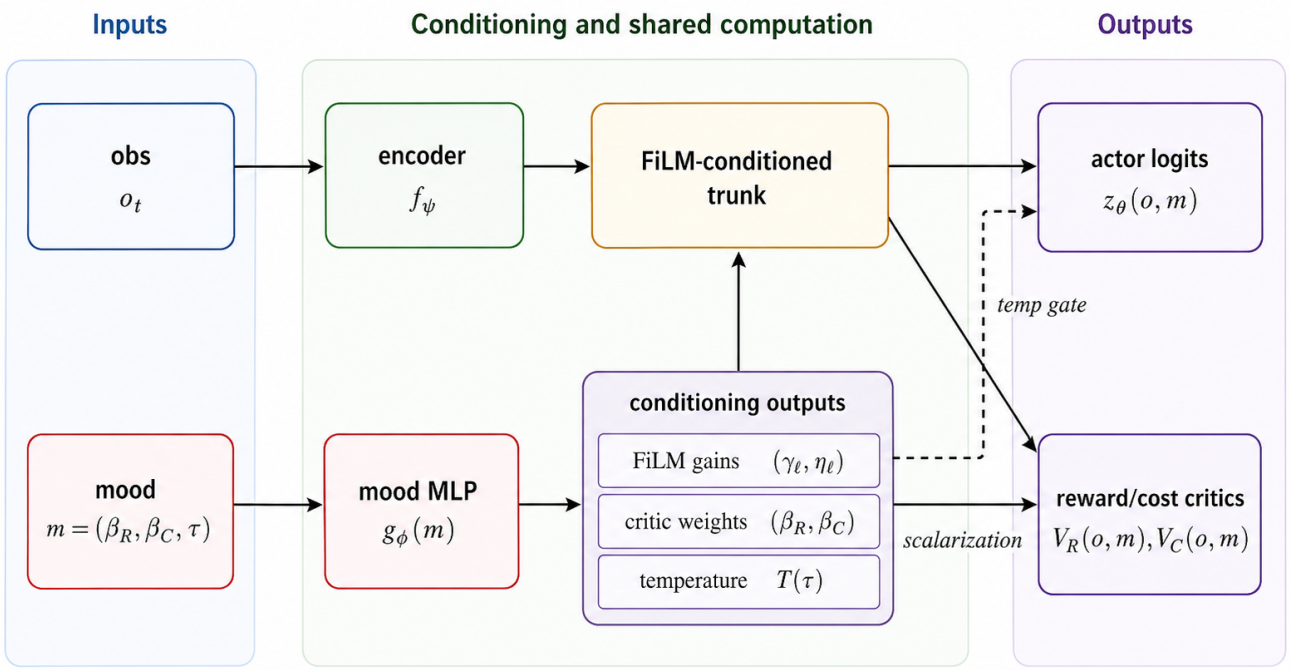

Figure 1: Mood-conditioned actor-critic. The observation is encoded once. The mood vector $m = (\beta_R, \beta_C, \tau)$ generates FiLM scales and shifts for the trunk, weights the decomposed reward-cost critic, and controls the final action temperature. At deployment, the controller changes only m; all network parameters remain fixed.

5 Experiments

Pac-Mind++. Pac-Mind++ is a 20×20 partially observable maze with walls, pellets, power pellets, and four moving ghosts. The agent observes a 15×15 egocentric crop with channels for walls, consumables, self position, ghost positions, ghost vulnerability, and recent motion. Pellets give $+1$, power pellets give $+10$, vulnerable ghost capture gives $+25$, clearing the maze gives $+100$, and each step gives -0.01. Collision with an active ghost terminates the episode and records unit cost. A proximity cost $0.05 \max(0, 3 - d)/3$ is added when the nearest active ghost is within Manhattan distance $d \leq 3$. Episodes terminate after collision, clearing the maze, or 3000 steps.

MiniHack-HazardRooms. MiniHack-HazardRooms uses the MiniHack sandbox built on the NetHack Learning Environment [Samvelyan et al., 2021, Kuttler et al., 2020]. Each episode samples one of 64 training layouts or 32 held-out layouts. Rooms contain lava, spikes, monsters, keys, doors, and an amulet goal. Reaching the amulet gives $+100$, useful pickups give $+5$, and steps give -0.01. Lava death and monster death record unit cost. Spike contact records cost 0.2 and does not always terminate. Observations are egocentric symbolic tensors.

Table 1: Evaluation design. Train and test splits are disjoint in layout seed and mood seed.

Domain	Train	Held-out
Pac-Mind++	128 layouts	64 layouts
MiniHack-HazardRooms	64 layouts	32 layouts
Safety-Gymnasium pilot	10 seeds	10 seeds

Mood distribution. Training samples

$$\beta_R \sim \text{LogUniform}(0.5, 3.0), \tag{19}$$
$$\beta_C \sim \text{LogUniform}(0.05, 3.0), \tag{20}$$
$$\tau \sim \text{Uniform}(0, 1). \tag{21}$$

With probability 0.2, training uses one of four named anchor moods: neutral $(1, 1, 0.2)$, seeking $(2, 0.25, 0.2)$, cautious $(1, 2, 0.3)$, or exploratory $(1, 0.5, 0.8)$. These anchors support qualitative probes. The specialist baseline is different: it trains 64 policies on a $4 \times 4 \times 4$ mood grid that includes the anchors.

Metrics and protocol. Main return and cost numbers use the stochastic policy with the deployed temperature. We also report entropy on the same observation buffer, before any greedy diagnostic. Held-out mood error is the normalized

mean absolute error between observed outcomes and multilinear interpolation from neighboring grid moods. Runtime sensitivity is measured on a common observation buffer so it is not confounded with different state visitation. Hypervolume uses return and negative cost after min-max normalization within each environment.

Each method is trained for 50M environment steps on Pac-Mind++ and 100M on MiniHack-HazardRooms with 12 random seeds. PPO uses rollout length 128, minibatch size 1024, clipping $\epsilon = 0.2$, GAE $\lambda = 0.95$, discount $\gamma = 0.99$, Adam learning rate 2.5×10^{-4}, and gradient-norm clipping at 0.5. Evaluation freezes weights and uses 512 episodes per seed and mood. Intervals are stratified bootstrap 95% intervals over seeds and layouts.

Baselines. We compare with neutral PPO, fixed-limit Lagrangian PPO, temperature-only PPO, concatenation-conditioned PPO, FiLM-conditioned PPO without vector critics, the 64-specialist library, and loss-only mood modulation. The loss-only baseline multiplies TD error, entropy coefficient, and risk penalty by mood-dependent gains during training, but the deployed actor is not conditioned on m.

6 Results

6.1 A single policy covers a reward-risk surface

Sweeping m after training reveals a structured frontier in both domains. Increasing β_R favors shorter routes and faster reward collection. Increasing β_C reduces contact with ghosts, lava, and monsters, usually by lengthening paths or waiting for hazards to move. Increasing τ raises entropy and helps sparse-goal discovery at moderate values; in narrow passages, high temperature increases dithering.

Figure 2 shows the resulting frontiers. MC-PPO dominates neutral PPO and fixed-limit Lagrangian PPO because it supplies many operating points with one set of weights. It approaches the specialist library at the library's trained moods while covering intermediate preferences with smooth changes in action distributions. The loss-only baseline has variation across separately trained checkpoints, but a sweep of a single frozen checkpoint collapses to one small cluster.

6.2 Held-out moods behave predictably

A useful control input should work between named settings. We evaluate 96 Latin-hypercube moods that are disjoint from the anchor set. MC-PPO places those moods between nearby evaluated grid points in both return and cost. The representative held-out return plot in Figure 3 has mean absolute error 0.07; the aggregate normalized response error is 0.049 ± 0.010 across seeds and layouts. Concatenation conditioning is less reliable because several seeds learn to ignore parts of the mood vector.

The local KL curve in Figure 3 separates smoothness from collapse. Loss-only modulation has almost zero runtime sensitivity, as expected. Concatenation and FiLM produce nonzero separation, but FiLM yields smoother local changes because the mood gates intermediate features rather than entering only as extra input coordinates. MC-PPO has the largest pairwise separation between anchor moods while keeping local perturbations stable.

6.3 Hazard alarms can reroute behavior

The online-switching test changes mood inside an episode. We compute an alarm score from the cost critic, normalize it by a running median absolute deviation, and switch when the score exceeds a threshold. The controller interpolates over 10 steps from $m_{\text{seek}} = (2, 0.25, 0.2)$ to $m_{\text{safe}} = (1, 2, 0.4)$. No parameters are updated.

In Pac-Mind++, the switch increases median distance to the nearest active ghost from 2.4 to 5.8 cells and reduces post-alarm collision probability from 6.1% to 1.7%. In MiniHack-HazardRooms, lava and monster deaths after the alarm fall from 8.4% to 2.3%, with a 9.6% reduction in remaining reward return. Figure 4 shows a typical Pac-Mind++ episode: the pre-alarm path cuts close to the hazard, while the post-alarm path routes around the wall after entropy rises.

6.4 Ablations isolate the mechanism

Table 3 compares architectural variants. Concatenating m to the observation is enough to create some control, but three of twelve seeds collapse to nearly identical action distributions across moods. FiLM improves sensitivity by making the mood affect multiple hidden layers. Vector critics improve frontier quality because reward and cost prediction no longer compete for a single scalar target. The temperature gate is most visible along the entropy axis. The smoothness-separation regularizer removes the remaining collapsed seed and improves interpolation, but the main gain comes from conditioning the actor and preserving separate reward-cost value targets.

Table 3: Ablations in Pac-Mind++. Collapse means a seed with mean pairwise $D_{\text{JS}} < 0.02$ across the evaluation mood grid.

Variant	HV ↑	Sens. ↑	Collapse ↓
Concatenation only	0.72 ± 0.03	0.13 ± 0.02	3/12
FiLM trunk	0.80 ± 0.02	0.21 ± 0.02	1/12
FiLM + vector critics	0.86 ± 0.02	0.24 ± 0.03	1/12
+ temperature gate	0.88 ± 0.02	0.31 ± 0.03	1/12
Full MC-PPO	0.90 ± 0.02	0.29 ± 0.02	0/12

6.5 Critic maps show what the gains change

Aggregate scores do not reveal whether the axes have interpretable local effects. We therefore evaluate one frozen Pac-Mind++ policy on matched observation buffers and plot the scalar value $V_m = \beta_R V_R - \beta_C V_C$ with action probabilities. Figure 5 shows the same state under three moods. High reward gain raises values along short pellet paths. High cost gain creates a basin near the active ghost and shifts the route

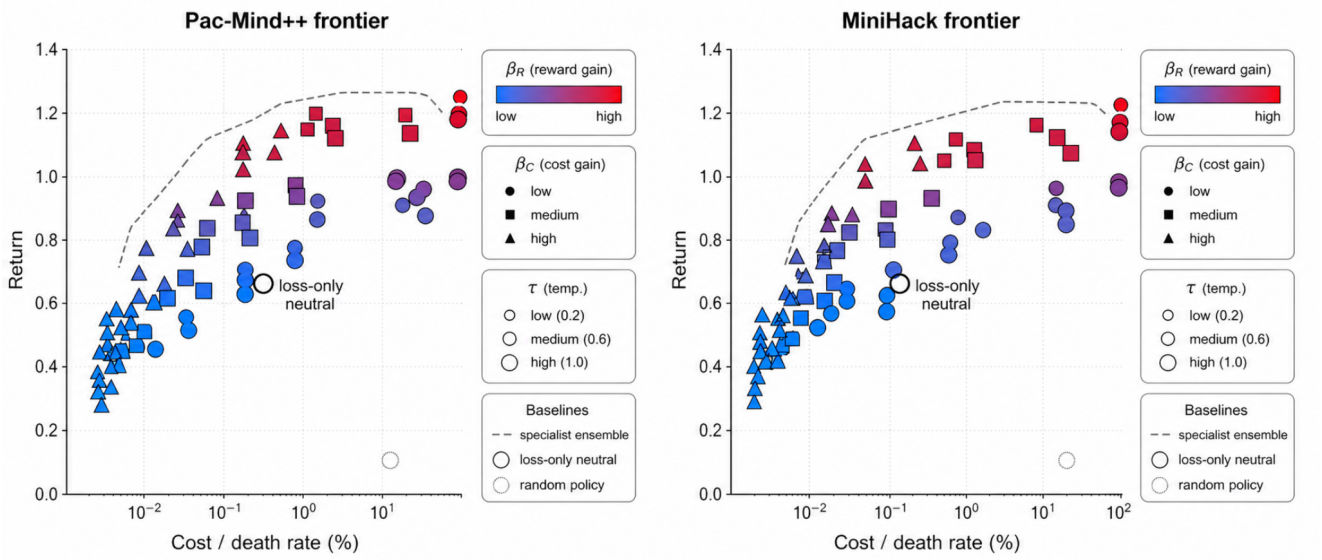

Figure 2: Reward-risk frontiers under deployment-time mood sweeps. Each point is one frozen policy evaluated at one mood. Color encodes reward gain β_R, marker shape encodes cost gain β_C, and marker size encodes temperature τ. MC-PPO covers a larger frontier than fixed-policy baselines and does not require the 64-policy specialist library.

Table 2: Main results. Hypervolume is normalized reward-cost frontier area. Held-out response error is the normalized mean absolute error for methods with a deployed mood surface. Post-alarm cost is the cost rate in the 100 steps after a hazard-triggered mood switch. Intervals are bootstrap 95% confidence intervals over seeds and layouts.

Method	Pac-Mind++ HV ↑	MiniHack HV ↑	Held-out response error ↓	Post-alarm cost ↓	Deployed policies
Neutral PPO	0.36 ± 0.03	0.31 ± 0.04	–	0.064 ± 0.008	1
Loss-only mood modulation	0.39 ± 0.04	0.34 ± 0.03	–	0.061 ± 0.007	1
Temperature-only PPO	0.51 ± 0.04	0.46 ± 0.05	0.210 ± 0.032	0.055 ± 0.006	1
Lagrangian PPO, fixed limit	0.58 ± 0.04	0.54 ± 0.04	–	0.039 ± 0.006	1
Concat-conditioned PPO	0.72 ± 0.03	0.65 ± 0.04	0.108 ± 0.019	0.036 ± 0.006	1
FiLM-conditioned PPO	0.80 ± 0.02	0.74 ± 0.03	0.071 ± 0.014	0.025 ± 0.005	1
Specialist ensemble	0.85 ± 0.02	0.79 ± 0.03	–	0.030 ± 0.006	64
MC-PPO	$\mathbf{0.90 \pm 0.02}$	$\mathbf{0.83 \pm 0.03}$	$\mathbf{0.049 \pm 0.010}$	$\mathbf{0.017 \pm 0.004}$	1

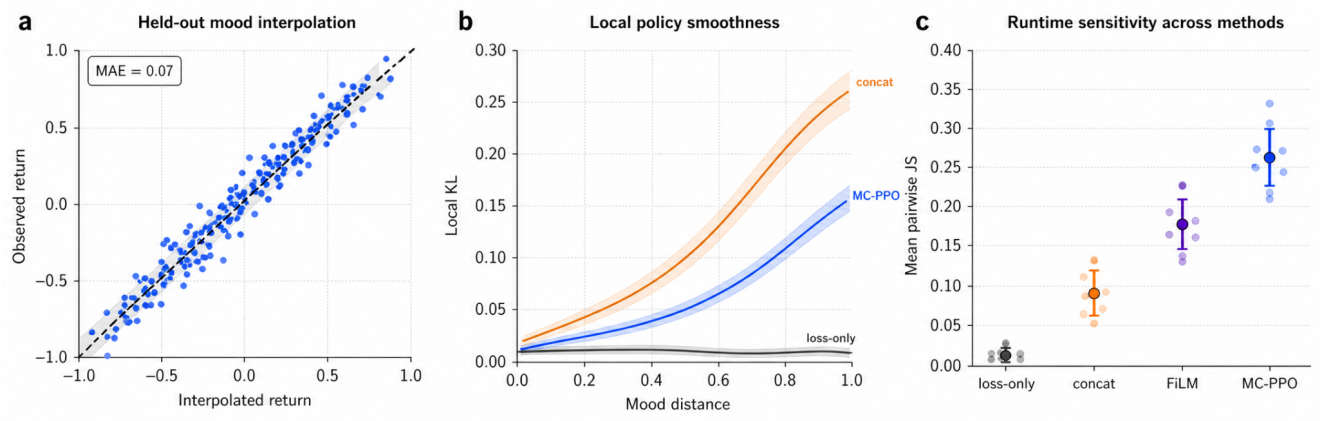

Figure 3: Interpolation and smoothness. Held-out moods fall between neighboring operating points for MC-PPO. Local KL grows smoothly for small mood perturbations, while distant moods remain behaviorally distinct. Loss-only modulation has almost no runtime sensitivity because the frozen policy does not read the mood.

around the wall. High temperature leaves the value geometry similar but spreads probability mass at junctions. These probes also catch failure modes: excessive cost gain can freeze the agent, and excessive temperature can destroy the precise action sequences needed in narrow corridors.

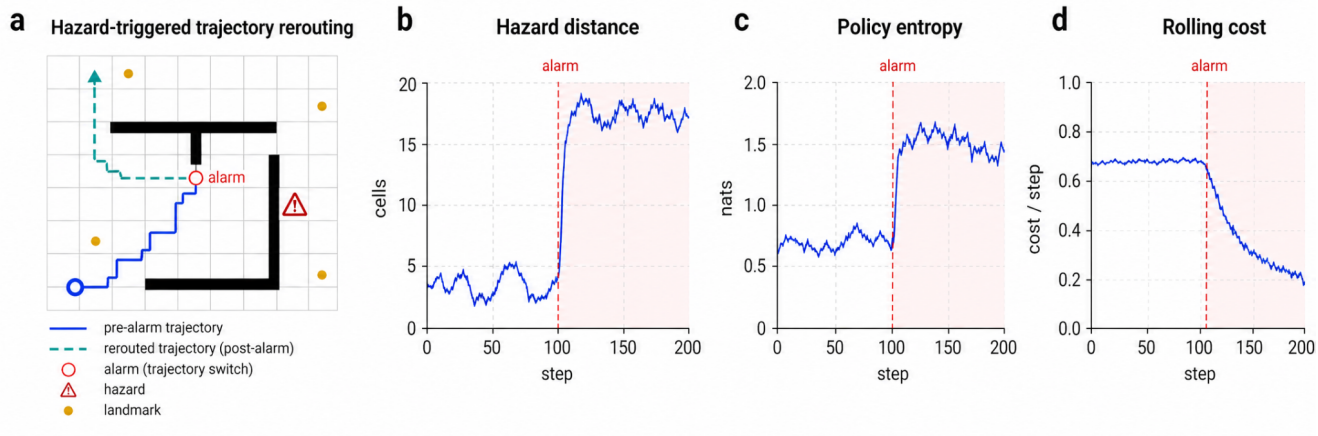

Figure 4: Hazard-triggered mood switching with frozen weights. The policy begins in a reward-seeking mood. When the cost critic raises an alarm, the controller moves to a cautious mood over 10 steps. The agent increases distance to hazards and reduces cost without retraining.

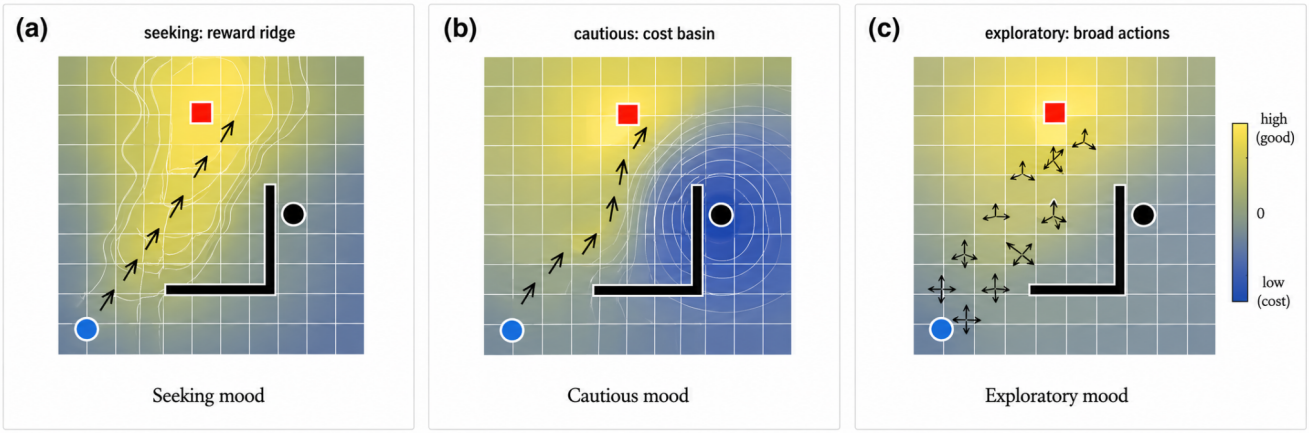

Figure 5: Critic and policy signatures for one frozen Pac-Mind++ policy. Color denotes scalar value V_m. Arrows denote action probabilities. The reward gain sharpens routes to pellets, the cost gain forms basins around active ghosts, and the temperature coordinate broadens action probabilities at junctions.

6.6 Specialists cover points, not a surface

The specialist library is strong at the moods it was trained for, but it is awkward as a runtime interface. Nearest-neighbor mood lookup creates discontinuities: in Pac-Mind++, crossing a Voronoi boundary between specialists produces a mean Jensen-Shannon action-distribution jump of 0.41 ± 0.05 on the shared observation buffer. MC-PPO's corresponding jump is 0.08 ± 0.02. A hazard alarm should bend the policy away from risk; swapping to an unrelated checkpoint can change unrelated habits at the same time.

The resource cost is also different. The 64-specialist library stores 64 actors and either loads policies on demand or evaluates a large batch to support rapid switching. MC-PPO adds 0.7% parameters for the mood encoder and FiLM projections. The forward-pass overhead is 3.1% in Pac-Mind++ and 2.4% in MiniHack-HazardRooms.

7 Discussion

The empirical claim is not that three numbers solve preference learning. It is that a policy can expose a stable behavioral surface after training, and that the surface is a better deployment object than either a single compromise policy or a shelf of specialists. The difference matters when context changes quickly. A supervisor can move along the surface with bounded steps, observe how return, cost, entropy, and local KL change, and keep the same representation, critic, and action conventions throughout the episode.

The method also exposes failure modes in a useful way. High τ creates exploration and can help with sparse rewards, but it also breaks precise action sequences in bottlenecks. High β_C protects the agent from hazards, but it can turn into waiting or freezing when the cost critic assigns broad risk to nearby states. High β_R recovers reward quickly and also encourages unsafe shortcuts when the cost channel is weak. These are not hidden surprises in separate checkpoints; they are regions of the same chart. That makes them candidates for hard bounds, hysteresis, or controller-side penalties.

There are limits to the formulation. Linear scalarization recovers supported trade-offs and may miss non-convex parts of a Pareto frontier. The cost function is assumed to be present, although many real systems need costs inferred from near misses, human feedback, or formal monitors. The environments here are discrete or symbolic; continuous-control systems may require different temperature parameterizations, especially under contact dynamics. The biological analogy should not be pushed past the computation it supports. Reward gain, cost gain, and temperature are controls chosen because RL policies already use them, not because they exhaust neuromodulator function.

The most direct extensions keep the interface but improve the signals behind it. Distributional critics could expose tail-risk coordinates such as CVaR. Successor-feature decompositions could support many reward and cost components without adding one critic per downstream task. Recurrent policies could add a memory-update coordinate, making adaptation rate a deployment input. A learned supervisor could choose m_t from uncertainty, energy budget, or formal risk estimates, while a safety layer restricts the admissible set.

8 Conclusion

A deployed RL system should not force every behavioral choice into the training run. The reward-risk trade-off that is right before a hazard appears is not the trade-off that is right after it appears, and a checkpoint that hides that distinction leaves the operator with poor options: retrain, swap policies, or accept a fixed compromise. MC-PPO gives the policy a different interface. The same frozen weights can be asked to seek reward, avoid cost, or widen exploration by changing three inputs that the actor and critic actually read.

That last clause is the core of the paper. Mood is not a label on a checkpoint and it is not a multiplier buried in the optimizer. It is a deployment variable only if rewriting it changes the forward pass that selects actions. Once stated this way, several design choices become necessary rather than cosmetic. The actor must be conditioned. The critic should keep reward and cost disentangled long enough to be reweighted. The temperature axis should have a monotone, inspectable effect on the action distribution. The evaluation has to sweep frozen policies and measure action-distribution changes on shared observations.

The experiments show what this buys. One policy covers a broad frontier, reaches moods that were not separately trained, and changes course after an alarm without gradient updates. It does not dominate specialists at every point, and it should not be deployed without bounds on admissible gains. Its advantage is that the trade-off is continuous, inspectable, and cheap to retarget. A specialist library can store many answers; a conditioned policy exposes the question that those answers vary with.

For reinforcement learning systems, this is a shift in what gets learned. The goal is not only to learn a behavior, but to learn a coordinate system over behaviors that remains usable after training. If that coordinate system is calibrated, a higher-level controller can make local decisions about risk and reward without reopening the optimizer. That is the practical content of the neuromodulatory analogy: broad, low-bandwidth signals can retune a policy when they are part of the policy's computation.

The work also suggests a stricter standard for evaluating adaptive RL interfaces. A claimed control variable should be tested after weights are frozen, on observations shared across methods, with both outcome metrics and action-distribution metrics. It should be swept beyond a few named settings, and interpolation should be measured rather than assumed. Under that standard, a loss coefficient, a post-hoc policy label, and a deployment input are different objects. Conflating them makes an agent look more controllable than it is.

The long-term value is not tied to these three axes. Other tasks may need energy, latency, fairness, uncertainty, or tail-risk coordinates. The design constraint is the same: the coordinates must be bounded, measurable, and connected to the action map.

References

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, pages 22–31. PMLR, 2017.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C. Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, volume 34, pages 29304–29320, 2021.

Andre Barreto, Will Dabney, Remi Munos, Jonathan J. Hunt, Tom Schaul, Hado van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Nathaniel D. Daw, Sham M. Kakade, and Peter Dayan. Opponent interactions between serotonin and dopamine. *Neural Networks*, 15(4–6):603–616, 2002.

Kenji Doya. Metalearning and neuromodulation. *Neural Networks*, 15(4–6):495–506, 2002.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, pages 1861–1870. PMLR, 2018.

Conor F. Hayes, Roxana Radulescu, Eugenio Bargiacchi, Roxana Kadar, Andrei Marinescu, Mathieu Reymond, Diederik M. Roijers, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirv Irissappane, Patrick Mannion, Ann Nowe, Gabriel Ramos, Marcello Restelli, and Peter Vamplew. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36(1):26, 2022.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Jiaming Ji, Borong Zhang, Jiayi Zhou, Xuehai Pan, Weidong Huang, Ruiyang Sun, Yiran Geng, Yifan Zhong, Juntao Dai, and Yaodong Yang. Safety-gymnasium: A unified safe reinforcement learning benchmark. arXiv preprint arXiv:2310.12567, 2023.

Heinrich Kuttler, Nantas Nardelli, Alexander H. Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktaschel. The NetHack Learning Environment. In *Advances in Neural Information Processing Systems*, volume 33, pages 7671–7684, 2020.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Mathieu Reymond, Eugenio Bargiacchi, and Ann Nowe. Pareto conditioned networks. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, pages 1110–1118, 2022.

Mikayel Samvelyan, Robert Kirk, Vitaly Kurin, Jack Parker-Holder, Minqi Jiang, Eric Hambro, Fabio Petroni, Heinrich Kuttler, Edward Grefenstette, and Tim Rocktaschel. MiniHack the planet: A sandbox for open-ended reinforcement learning research. In *Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1312–1320. PMLR, 2015.

Wolfram Schultz, Peter Dayan, and P. Read Montague. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997.

A Proofs

A.1 Proof of Proposition 1

For a fixed mood, define the scalar reward $\tilde{r}_t = \beta_R r_t - \beta_C c_t$. The policy-gradient theorem gives

$$\nabla_\theta J_{\tilde{r}}(\theta; m) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a_t \mid o_t, m) A^m_{\tilde{r}}(o_t, a_t)\right]. \tag{22}$$

Linearity of expectation gives $Q^m_{\tilde{r}} = \beta_R Q^m_R - \beta_C Q^m_C$ and $V^m_{\tilde{r}} = \beta_R V^m_R - \beta_C V^m_C$. Thus $A^m_{\tilde{r}} = \beta_R A^m_R - \beta_C A^m_C$. The entropy term adds the standard entropy-gradient contribution.

A.2 Proof of Proposition 2

For a fixed policy π,

$$J_\pi(m) = \beta_R J^\pi_R - \beta_C J^\pi_C + \tau J^\pi_H. \tag{23}$$

Bounded rewards, costs, and entropy imply

$$|J^\pi_R| \le \frac{R_{\max}}{1-\gamma}, \quad |J^\pi_C| \le \frac{C_{\max}}{1-\gamma}, \quad |J^\pi_H| \le \frac{\log|\mathcal{A}|}{1-\gamma}. \tag{24}$$

Applying the triangle inequality to $J_\pi(m) - J_\pi(m')$ yields the claimed bound.

A.3 Proof of Proposition 3

If the deployed computation is $\pi_\theta(a \mid o)$, the probabilities are functions only of o and fixed parameters θ. Setting an external variable m cannot change a computation that never reads it, so $\pi_\theta(\cdot \mid o, m) = \pi_\theta(\cdot \mid o, m')$ for all moods.

B Implementation Details

Pac-Mind++ uses a $15 \times 15 \times 8$ egocentric tensor. The encoder has three convolutional layers with 32, 64, and 64 channels, kernel size 3, stride 1, and ReLU activations. A two-layer MLP maps the flattened representation to a 512-dimensional trunk. MiniHack-HazardRooms uses the same trunk after replacing raw glyphs with learned embeddings for glyph id, color, and special terrain flags. The mood encoder is a two-layer MLP with hidden width 64 and SiLU activations. FiLM parameters are produced for the final two trunk layers.

Table 4: Default hyperparameters.

Parameter	Value
Discount γ	0.99
GAE λ	0.95
PPO clip ϵ	0.2
Rollout length	128
Parallel environments	64
Minibatch size	1024
PPO epochs per rollout	4
Adam learning rate	2.5×10^{-4}
Gradient clip norm	0.5
Mood smoothness weight	0.01
Mood separation weight	0.005

C Environment Specifications

Pac-Mind++. The maze generator samples a connected wall layout from a randomized depth-first-search template, then inserts loops until the average shortest path between free cells falls below 35. Pellets are placed on 35% of free cells. Four power pellets are placed in quadrants. Ghosts use a mixed policy: with probability 0.75 they reduce shortest-path distance to the agent, with probability 0.15 they move randomly, and with probability 0.10 they continue their previous direction if legal. Power pellets make ghosts vulnerable for 40 steps.

MiniHack-HazardRooms. HazardRooms uses room templates with variable door placement, lava pools, spike corridors, monsters, and optional keys. Training uses 64 layout seeds. Evaluation uses 32 disjoint layout seeds and random object placements. Lava death and monster death are terminal. Spikes create cost 0.2 and reward penalty -10 but are nonterminal with probability 0.7. The amulet is reachable in all generated layouts, and a shortest safe path exists by construction.

D Additional Results

Table 5: Anchor moods in Pac-Mind++. Return and cost are means over 12 seeds. Entropy is measured before greedy diagnostics, using the stochastic policy.

Mood	β_R	β_C	τ	Return ↑	Collision % ↓
Neutral	1.0	1.0	0.2	141 ± 4	2.4 ± 0.3
Seeking	2.0	0.25	0.2	167 ± 5	5.8 ± 0.5
Cautious	1.0	2.0	0.3	132 ± 3	0.9 ± 0.2
Exploratory	1.0	0.5	0.8	148 ± 6	3.1 ± 0.4
High reward, high caution	2.0	2.0	0.3	158 ± 4	1.6 ± 0.3
Low reward, high caution	0.5	3.0	0.2	96 ± 3	0.4 ± 0.1

Table 6: Safety-Gymnasium pilot. Values are normalized across methods.

Method	PointGoal HV ↑	CarGoal HV ↑
Neutral PPO	0.42 ± 0.05	0.37 ± 0.06
Lagrangian PPO	0.61 ± 0.04	0.55 ± 0.05
Concat-conditioned PPO	0.69 ± 0.04	0.60 ± 0.05
MC-PPO	0.78 ± 0.03	0.68 ± 0.04

E Failure Modes

The control surface has three recurring failure modes. High τ can produce dithering when the optimal path requires precise moves through a bottleneck, especially in MiniHack lava corridors. Very large β_C can freeze the agent: it preserves safety but stops collecting reward because most nearby states are assigned high expected cost. Very large β_R with low β_C creates unsafe shortcuts. These failures are predictable from the axes and can be used to set admissible deployment ranges.